Postel Home

Network Working Group
Request for Comments: 917

Jeffrey Mogul
Computer Science Department
Stanford University
October 1984

INTERNET SUBNETS

Status Of This Memo

This RFC suggests a proposed protocol for the ARPA-Internet community, and requests discussion and suggestions for improvements. Distribution of this memo is unlimited.

Overview

We discuss the utility of "subnets" of Internet networks, which are logically visible sub-sections of a single Internet network. For administrative or technical reasons, many organizations have chosen to divide one Internet network into several subnets, instead of acquiring a set of Internet network numbers.

We propose procedures for the use of subnets, and discuss approaches to solving the problems that arise, particularly that of routing.

Acknowledgment

This proposal is the result of discussion with several other people. J. Noel Chiappa, Chris Kent, and Tim Mann, in particular, provided important suggestions.

1. Introduction

The original view of the Internet universe was a two-level hierarchy: the top level the catenet as a whole, and the level below it a collection of "Internet Networks", each with its own Network Number. (We do not mean that the Internet has a hierarchical topology, but that the interpretation of addresses is hierarchical.)

While this view has proved simple and powerful, a number of organizations have found it inadequate and have added a third level to the interpretation of Internet addresses. In this view, a given Internet Network might (or might not) be divided into a collection of subnets.

The original, two-level, view carries a strong presumption that, to a host on an Internet network, that network may be viewed as a single edge; to put it another way, the network may be treated as a "black box" to which a set of hosts is connected. This is true of the

ARPANET, because the IMPs mask the use of specific links in that network. It is also true of most local area network (LAN) technologies, such as Ethernet or ring networks.

However, this presumption fails in many practical cases, because in moderately large organizations (e.g., Universities or companies with more than one building) it is often necessary to use more than one LAN cable to cover a "local area". For example, at this writing there are eighteen such cables in use at Stanford University, with more planned.

There are several reasons why an organization might use more than one cable to cover a campus:

- Different technologies: Especially in a research environment, there may be more than one kind of LAN in use; e.g., an organization may have some equipment that supports Ethernet, and some that supports a ring network.

- Limits of technologies: Most LAN technologies impose limits, based electrical parameters, on the number of hosts connected, and on the total length of the cable. It is easy to exceed these limits, especially those on cable length.

- Network congestion: It is possible for a small subset of the hosts on a LAN to monopolize most of the bandwidth. A common solution to this problem is to divide the hosts into cliques of high mutual communication, and put these cliques on separate cables.

- Point-to-Point links: Sometimes a "local area", such as a university campus, is split into two locations too far apart to connect using the preferred LAN technology. In this case, high-speed point-to-point links might connect several LANs.

An organization that has been forced to use more than one LAN has three choices for assigning Internet addresses:

1. Acquire a distinct Internet network number for each cable.

2. Use a single network number for the entire organization, but assign host numbers without regard to which LAN a host is on. (We will call this choice "transparent subnets".)

3. Use a single network number, and partition the host address space by assigning subnet numbers to the LANs. ("Explicit subnets".)

Each of these approaches has disadvantages. The first, although not requiring any new or modified protocols, does result in an explosion in the size of Internet routing tables. Information about the internal details of local connectivity is propagated everywhere, although it is of little or no use outside the local organization. Especially as some current gateway implementations do not have much space for routing tables, it would be nice to avoid this problem.

The second approach requires some convention or protocol that makes the collection of LANs appear to be a single Internet network. For example, this can be done on LANs where each Internet address is translated to a hardware address using an Address Resolution Protocol (ARP), by having the bridges between the LANs intercept ARP requests for non-local targets. However, it is not possible to do this for all LAN technologies, especially those where ARP protocols are not currently used, or if the LAN does not support broadcasts. A more fundamental problem is that bridges must discover which LAN a host is on, perhaps by using a broadcast algorithm. As the number of LANs grows, the cost of broadcasting grows as well; also, the size of translation caches required in the bridges grows with the total number of hosts in the network.

The third approach addresses the key problem: existing standards assume that all hosts on an Internet local network are on a single cable. The solution is to explicitly support subnets. This does have a disadvantage, in that it is a modification of the Internet Protocol, and thus requires changes to IP implementations already in use (if these implementations are to be used on a subnetted network.) However, we believe that these changes are relatively minor, and once made, yield a simple and efficient solution to the problem. Also, the approach we take in this document is to avoid any changes that would be incompatible with existing hosts on non-subnetted networks.

Further, when appropriate design choices are made, it is possible for hosts which believe they are on a non-subnetted network to be used on a subnetted one, as will be explained later. This is useful when it is not possible to modify some of the hosts to support subnets explicitly, or when a gradual transition is preferred. Because of this, there seems little reason to use the second approach listed above.

The rest of this document describes approaches to subnets of Internet Networks.

1.1. Terminology

To avoid either ambiguity or prolixity, we will define a few terms, which will be used in the following sections:

Catenet

> The collection of connected Internet Networks

Network

> A single Internet network (that may or may not be divided into subnets.)

Subnet

> A subnet of an Internet network.

Network Number

> As in [8].

Local Address

> The bits in an Internet address not used for the network number; also known as "rest field".

Subnet Number

> A number identifying a subnet within a network.

Subnet Field

> The bit field in an Internet address used for the subnet number.

Host Field

> The bit field in an Internet address used for denoting a specific host.

Gateway

> A node connected to two or more administratively distinct networks and/or subnets, to which hosts send datagrams to be forwarded.

Bridge

A node connected to two or more administratively indistinguishable but physically distinct subnets, that automatically forwards datagrams when necessary, but whose existence is not know to other hosts. Also called a "software repeater".

2. Standards for Subnet Addressing

Following the division presented in [2], we observe that subnets are fundamentally an issue of addressing. In this section, we first describe a proposal for interpretation of Internet Addressing to support subnets. We then discuss the interaction between this address format and broadcasting; finally, we present a protocol for discovering what address interpretation is in use on a given network.

2.1. Interpretation of Internet Addresses

Suppose that an organization has been assigned an Internet network number, has further divided that network into a set of subnets, and wants to assign host addresses: how should this be done? Since there are minimal restrictions on the assignment of the "local address" part of the Internet address, several approaches have been proposed for representing the subnet number:

1. Variable-width field: Any number of the bits of the local address part are used for the subnet number; the size of this field, although constant for a given network, varies from network to network. If the field width is zero, then subnets are not in use.

2. Fixed-width field: A specific number of bits (e.g., eight) is used for the subnet number, if subnets are in use.

3. Self-encoding variable-width field: Just as the width (i.e., class) of the network number field is encoded by its high-order bits, the width of the subnet field is similarly encoded.

4. Self-encoding fixed-width field: A specific number of bits is is used for the subnet number. Subnets are in use if the high-order bit of this field is one; otherwise, the entire local address part is used for host number.

Since there seems to be no advantage in doing otherwise, all these schemes place the subnet field as the most significant field in

the local address part. Also, since the local address part of a Class C address is so small, there is little reason to support subnets of other than Class A and Class B networks.

What criteria can we use to choose one of these four schemes? First, do we want to use a self-encoding scheme; that is, should it be possible to tell from examining an Internet address if it refers to a subnetted network, without reference to any other information?

One advantage to self-encoding is that it allows one to determine if a non-local network has been divided into subnets. It is not clear that this would be of any use. The principle advantage, however, is that no additional information is needed for an implementation to determine if two addresses are on the same subnet. However, this can also be viewed as a disadvantage: it may cause problems for non-subnetted networks which have existing host numbers that use arbitrary bits in the local address part <1>. In other words, it is useful to be able control whether a network is subnetted independently from the assignment of host addresses. Another disadvantage of any self-encoding scheme is that it reduces the local address space by at least a factor of two.

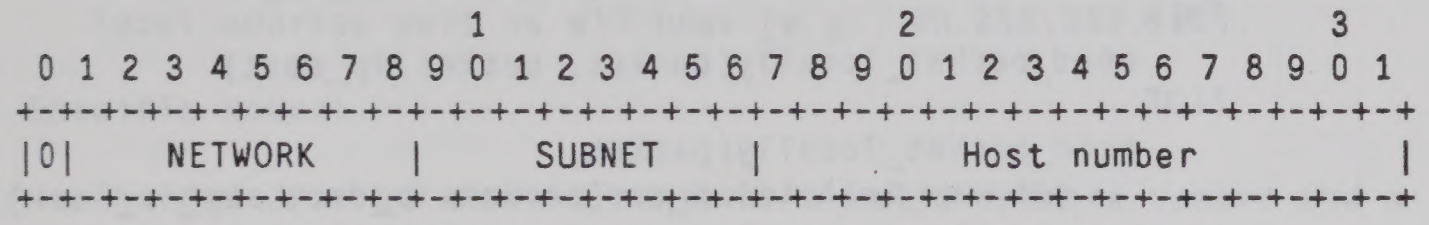
If a self-encoding scheme is not used, it is clear that a variable-width subnet field is appropriate. Since there must in any case be some per-network "flag" to indicate if subnets are in use, the additional cost of using an integer (the subnet field width) instead of a boolean is negligible. The advantage of using a variable-width subnet field is that it allows each organization to choose the best way to allocate relatively scarce bits of local address to subnet and host numbers.

Our proposal, therefore, is that the Internet address be interpreted as:

<network-number><subnet-number><host-number>

where the <network-number> field is as in [8], the <host-number> field is at least one bit wide, and the width of the <subnet-number> field is constant for a given network. No further structure is required for the <subnet-number> or <host-number> fields. If the width of the <subnet-number> field is zero, then the network is not subnetted (i.e., the interpretation of [8] is used.)

For example, on a Class A network with an eight bit wide subnet field, an address is broken down like this:



We expect that, for reasons of simplicity and efficient implementation, that most organizations will choose a subnet field width that is a multiple of eight bits. However, an implementation must be prepared to handle other possible widths.

We reject the use of "recursive subnets", the division of the host field into "sub-subnet" and host parts, because:

- There is no obvious need for a four-level hierarchy.

- The number of bits available in an IP address is not large enough to make this useful in general.

- The extra mechanism required is complex.

2.2. Changes to Host Software to Support Subnets

In most implementations of IP, there is code in the module that handles outgoing packet that does something like:

```
IF ip_net_number(packet.ip_dest) = ip_net_number(my_ip_addr)
    THEN
        send_packet_locally(packet, packet.ip_dest)
    ELSE
        send_packet_locally(packet,
            gateway_to(ip_net_number(packet.ip_dest)))
```

(If the code supports multiple connected networks, it will be more complicated, but this is irrelevant to the current discussion.)

To support subnets, it is necessary to store one more 32-bit quantity, called my_ip_mask. This is a bit-mask with bits set in the fields corresponding to the IP network number, and additional bits set corresponding to the subnet number field. For example, on a Class A network using an eight-bit wide subnet field, the mask would be 255.255.0.0.

The code then becomes:

```
IF bitwise_and(packet.ip_dest, my_ip_mask)
                = bitwise_and(my_ip_addr, my_ip_mask)
    THEN
        send_packet_locally(packet, packet.ip_dest)
    ELSE
        send_packet_locally(packet,
           gateway_to(bitwise_and(packet.ip_dest, my_ip_mask)))
```

Of course, part of the expression in the conditionally can be pre-computed.

It may or may not be necessary to modify the "gateway_to" function, so that it performs comparisons in the same way.

To support multiply-connected hosts, the code can be changed to keep the "my_ip_addr" and "my_ip_mask" quantities on a per-interface basis; the expression in the conditional must then be evaluated for each interface.

2.3. Subnets and Broadcasting

In the absence of subnets, there are only two kinds of broadcast possible within the Internet Protocol <2>: broadcast to all hosts on a specific network, or broadcast to all hosts on "this network"; the latter is useful when a host does not know what network it is on.

When subnets are used, the situation becomes slightly more complicated. First, the possibility now exists of broadcasting to a specific subnet. Second, broadcasting to all the hosts on a subnetted network requires additional mechanism; in [6] the use of "Reverse Path Forwarding" [3] is proposed. Finally, the interpretation of a broadcast to "this network" is that it should not be forwarded outside of the original subnet.

Implementations must therefore recognize three kinds of broadcast addresses, in addition to their own host addresses:

This physical network

> A destination address of all ones (255.255.255.255) causes the a datagram to be sent as a broadcast on the local physical network; it must not be forwarded by any gateway.

Specific network

The destination address contains a valid network number; the local address part is all ones (e.g., 36.255.255.255).

Specific subnet

The destination address contains a valid network number and a valid subnet number; the host field is all ones (e.g., 36.40.255.255).

For further discussion of Internet broadcasting, see [6].

One factor that may aid in deciding whether to use subnets is that it is possible to broadcast to all hosts of a subnetted network with a single operation at the originating host. It is not possible to broadcast, in one step, to the same set of hosts if they are on distinct networks.

2.4. Determining the Width of the Subnet Field

How can a host (or gateway) determine what subnet field width is in use on a network to which it is connected? The problem is analogous to several other "bootstrapping" problems for Internet hosts: how a host determines its own address, and how it locates a gateway on its local network. In all three cases, there are two basic solutions: "hardwired" information, and broadcast-based protocols.

"Hardwired" information is that available to a host in isolation from a network. It may be compiled-in, or (preferably) stored in a disk file. However, for the increasingly common case of a diskless workstation that is bootloaded over a LAN, neither hard-wired solution is satisfactory. Instead, since most LAN technology supports broadcasting, a better method is for the newly-booted host to broadcast a request for the necessary information. For example, for the purpose of determining its Internet address, a host may use the "Reverse Address Resolution Protocol" [4].

We propose to extend the ICMP protocol [9] by adding a new pair of ICMP message types, "Address Format Request" and "Address Format Reply", analogous to the "Information Request" and "Information Reply" ICMP messages. These are described in detail in Appendix I.

The intended use of these new ICMPs is that a host, when booting,

broadcast an "Address Format Request" message <3>. A gateway (or a host acting in lieu of a gateway) that receives this message responds with an "Address Format Reply". If there is no indication in the request which host sent it (i.e., the IP Source Address is zero), the reply is broadcast as well. The requesting host will hear the response, and from it determine the width of the subnet field.

Since there is only one possible value that can be sent in an "Address Format Reply" on any given LAN, there is no need for the requesting host to match the responses it hears against the request it sent; similarly, there is no problem if more than one gateway responds. We assume that hosts reboot infrequently, so the broadcast load on a network from use of this protocol should be small.

If a host is connected to more than one LAN, it must use this protocol on each, unless it can determine (from a response on one of the LANs) that several of the LANs are part of the same network, and thus must have the same subnet field width.

One potential problem is what a host should do if it receives no response to its "Address Format Request", even after a reasonable number of tries. Three interpretations can be placed on the situation:

1. The local net exists in (permanent) isolation from all other nets.

2. Subnets are not in use, and no host supports this ICMP request.

3. All gateways on the local net are (temporarily) down.

The first and second situations imply that the subnet field width is zero. In the third situation, there is no way to determine what the proper value is; the safest choice is thus zero. Although this might later turn out to be wrong, it will not prevent transmissions that would otherwise succeed. It is possible for a host to recover from a wrong choice: when a gateway comes up, it should broadcast an "Address Format Reply"; when a host receives such a message that disagrees with its guess, it should adjust its data structures to conform to the received value. No host or gateway should send an "Address Format Reply" based on a "guessed" value.

Finally, note that no host is required to use this ICMP protocol to discover the subnet field width; it is perfectly reasonable for a host with non-volatile storage to use stored information.

3. Subnet Routing Methods

One problem that faces all Internet hosts is how to determine a route to another host. In the presence of subnets, this problem is only slightly modified.

The use of subnets means that there are two levels to the routing process, instead of one. If the destination host is on the same network as the source host, the routing decision involves only the subnet gateways between the hosts. If the destination is on a different network, then the routing decision requires the choice both of a gateway out of the source host's network, and of a route within the network to that gateway.

Fortunately, many hosts can ignore this distinction (and, in fact, ignore all routing choices) by using a "default" gateway as the initial route to all destinations, and relying on ICMP Host Redirect messages to define more appropriate routes. However, this is not an efficient method for a gateway or for a multi-homed host, since a redirect may not make up for a poor initial choice of route. Such hosts should use a routing information exchange protocol, but that is beyond the scope of this document; in any case, the problem arises even when subnets are not used.

The problem for a singly-connected host is thus to find at least one neighbor gateway. Again, there are basic two solutions to this: use hard-wired information, or use broadcasts. We believe that the neighbor-gateway acquisition problem is the same with or without subnets, and thus the choice of solution is not affected by the use of subnets.

However, one problem remains: a source host must determine if datagram to a given destination address must be sent via a gateway, or sent directly to the destination host. In other words, is the destination host on the same physical network as the source? This particular phase of the routing process is the only one that requires an implementation to be explicitly aware of subnets; in fact, if broadcasts are not used, it is the only place where an Internet implementation must be modified to support subnets.

Because of this, it is possible to use some existing implementations without modification in the presence of subnets <4>. For this to work, such implementations must:

- Be used only on singly-homed hosts, and not as a gateway.

- Be used on a broadcast LAN.

- Use an Address Resolution Protocol (ARP), such [7].

- Not be required to maintain connections in the case of gateway crashes.

In this case, one can modify the ARP server module in a subnet gateway so that when it receives an ARP request, it checks the target Internet address to see if it is along the best route to the target. If it is, it sends to the requesting host an ARP response indicating its own hardware address. The requesting host thus believes that it knows the hardware address of the destination host, and sends packets to that address. In fact, the packets are received by the gateway, and forwarded to the destination host by the usual means.

This method requires some blurring of the layers in the gateways, since the ARP server and the Internet routing table would normally not have any contact. In this respect, it is somewhat unsatisfactory. Still, it is fairly easy to implement, and does not have significant performance costs. One problem is that if the original gateway crashes, there is no way for the source host to choose an alternate route even if one exists; thus, a connection that might otherwise have been maintained will be broken.

One should not confuse this method of "ARP-based subnetting" with the superficially similar use of ARP-based bridges. ARP-based subnetting is based on the ability of a gateway to examine an IP address and deduce a route to the destination, based on explicit subnet topology. In other words, a small part of the routing decision has been moved from the source host into the gateway. An ARP-based bridge, in contrast, must somehow locate each host without any assistance from a mapping between host address and topology. Systems built out of ARP-based bridges should not be referred to as "subnetted".

N.B.: the use of ARP-based subnetting is complicated by the use of broadcasts. An ARP server [7] should never respond to a request whose target is a broadcast address. Such a request can only come from a host that does not recognize the broadcast address as such, and so honoring it would almost certainly lead to a forwarding loop. If there are N such hosts on the physical network that do not recognize this address as a broadcast, then a packet sent with a Time-To-Live of T could potentially give rise to T**N spurious re-broadcasts.

4. Case Studies

In this section, we briefly sketch how subnets have been used by several organizations.

4.1. Stanford University

At Stanford, subnets were introduced initially for historical reasons. Stanford had been using the Pup protocols [1] on a collection of several Experimental Ethernets [5] since 1979, several years before Internet protocols came into use. There were a number of Pup gateways in service, and all hosts and gateways acquired and exchanged routing table information using a simple broadcast protocol.

When the Internet Protocol was introduced, the decision was made to use an eight-bit wide subnet number; Internet subnet numbers were chosen to match the Pup network number of a given Ethernet, and the Pup host numbers (also eight bits) were used as the host field of the Internet address.

The Pup-only gateways were then modified to forward Internet datagrams according to their Pup routing tables; they otherwise had no understanding of Internet packets and in fact did not adjust the Time-to-live field in the Internet header. This seems to be acceptable, since bugs that caused forwarding loops have not appeared. The Internet hosts that are multi-homed and thus can serve as gateways do adjust the Time-to-live field; since all of the currently also serve as Pup gateways, no additional routing information exchange protocol was needed.

Internet host implementations were modified to understand subnets (in several different ways, but with identical effects). Since all already had Pup implementations, the Internet routing tables were maintained by the same process that maintained the Pup routing tables, simply translating the Pup network numbers into Internet subnet numbers.

When 10Mbit Ethernets were added, the gateways were modified to use the ARP-based scheme described in an earlier section; this allowed unmodified hosts to be used on the 10Mbit Ethernets.

IP subnets have been in use since early 1982; currently, there are about 330 hosts, 18 subnets, and a similar number of subnet gateways in service. Once the Pup-only gateways are converted to be true Internet gateways, an Internet-based routing exchange protocol will be introduced, and Pup will be phased out.

4.2. MIT

MIT was the first IP site to accumulate a large collection of local network links. Since this happened before network numbers were divided into classes, to have assigned each link at MIT its own IP network number would have used up a good portion of the available address space. MIT decided to use one IP network number, and to manage the 24-bit "rest" field itself, by dividing it into three 8-bit fields; "subnet", "reserved, must be zero", and "host". Since the CHAOS protocol already in use at MIT used an 8-bit subnet number field, it was possible to assign each link the same subnet number in both protocols. The IP host field was set to 8 bits since most available local net hardware at that point used 8 bit addresses, as did the CHAOS protocol; it was felt that reserving some bits for the future was wise.

The initial plan was to use a dynamic routing protocol between the IP subnet gateways; several such protocols have been mooted but nobody has bothered to implement one; static routing tables are still used. It is likely that this change will finally be made soon.

To solve the problem that imported IP software always needed modification to work in the subnetted environment, MIT searched for a model of operation that led to the least change in host IP software. This led to a model where IP gateways send ICMP Host Redirects rather than Network Redirects. All internal MIT IP gateways now do so. With hosts that can maintain IP routing tables for non-local communication on a per host basis, this hides most of the subnet structure. The "minimum adjustment" for host software to work correctly in both subnetted and non-subnetted environments is the bit-mask algorithm mentioned earlier.

MIT has no immediate plans to move toward a single "approved" protocol; this is due partly to the degree of local autonomy and the amount of installed software, and partly to the lack of a single prominent industry standard. Rather, the approach taken has been to provide a single set of physical links and packet switches, and to layer several "virtual" protocol nets atop the single set of links. MIT has had some bad experiences with trying to exchange routing information between protocols and wrap one protocol in another; the general approach is to keep the protocols strictly separated except for sharing the basic hardware. Using ARP to hide the subnet structure is not much in favor; it is felt that this overloads the address resolution operation. In a complicated system (i.e. one with loops, and variant link speeds),

a more sophisticated information interchange will be needed between gateways; making this an explicit mechanism (but one insulated from the hosts) was felt to be best.

4.3. Carnegie-Mellon University

CMU uses a Class B network currently divided into 11 physical subnets (two 3Mbit Experimental Ethernets, seven 10Mbit Ethernets, and two ProNet rings.) Although host numbers are assigned so that all addresses with a given third octet will be on the same subnet (but not necessarily vice versa), this is essentially an administrative convenience. No software currently knows the specifics of this allocation mechanism or depends on it to route between cables.

Instead, an ARP-based bridge scheme is used. When a host broadcasts an ARP request, all bridges which receive it cache the original protocol address mapping and then forward the request (after the appropriate adjustments) as an ARP broadcast request onto each of their other connected cables. When a bridge receives a non-broadcast ARP reply with a target protocol address not its own, it consults its ARP cache to determine the cable onto which the reply should be forwarded. The bridges thus attempt to transparently extend the ARP protocol into a heterogenous multi-cable environment. They are therefore required to turn ARP broadcasts on a single cable into ARP broadcasts on all other connected cables even when they "know better". This algorithm works only in the absence of cycles in the network connectivity graph (which is currently the case). Work is underway to replace this simple-minded algorithm with a protocol implemented among the bridges, in support of redundant paths and to reduce the collective broadcast load. The intent is to retain the ARP base and host transparency, if possible.

Implementations supporting the 3Mbit Ethernet and 10Mb proNET ring at CMU use RFC-826 ARP (instead of some wired-in mapping such as simply using the 8-bit hardware address as the the fourth octet of the IP address).

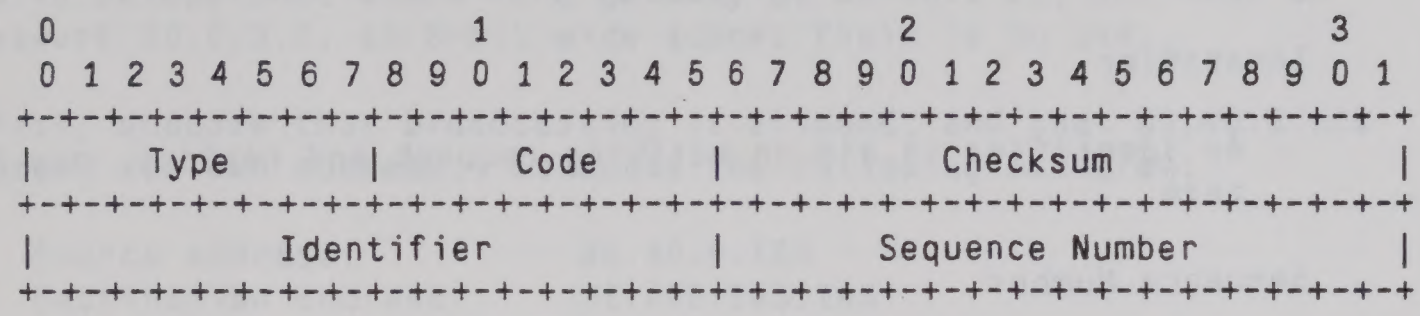
Since there are currently no redundant paths between cables, the issue of maintaining connections across bridge crashes is moot. With about 150 IP-capable hosts on the net, the bridge caches are still of reasonable size, and little bandwidth is devoted to ARP broadcast forwarding.

CMU's network is likely to grow from its relatively small, singly-connected configuration centered within their CS/RI

facility to a campus-wide intra-departmental configuration with 5000-10000 hosts and redundant connections between cables. It is possible that the ARP-based bridge scheme will not scale to this size, and a system of explicit subnets may be required. The medium-term goal, however, is an environment into which unmodified extant (especially 10Mb ethernet based) IP implementations can be imported; the intent is to stay with a host-transparent (thus ARP-based) routing mechanism as long as possible. CMU is concerned that even if subnets become part of the IP standard they will not be widely implemented; this is the major obstacle to their use at CMU.

I. Address Format ICMP

Address Format Request or Address Format Reply



IP Fields:

Addresses

The address of the source in an address format request message will be the destination of the address format reply message. To form an address format reply message, the source address of the request becomes the destination address of the reply, the source address of the reply is set to the replier's address, the type code changed to A2, the subnet field width inserted into the Code field, and the checksum recomputed. However, if the source address in the request message is zero, then the destination address for the reply message should denote a broadcast.

ICMP Fields:

Type

A1 for address format request message

A2 for address format reply message

Code

0 for address format request message

Width of subnet field, in bits, for address format reply message

Checksum

The checksum is the 16-bit one's complement of the one's

complement sum of the ICMP message starting with the ICMP Type. For computing the checksum, the checksum field should be zero. This checksum may be replaced in the future.

Identifier

An identifier to aid in matching request and replies, may be zero.

Sequence Number

A sequence number to aid in matching request and replies, may be zero.

Description

A gateway receiving an address format request should return it with the Code field set to the number of bits of Subnet number in IP addresses for the network to which the datagram was addressed. If the request was broadcast, the destination network is "this network". The Subnet field width may be from 0 to (31 - N), where N is the width in bits of the IP net number field (i.e., 8, 16, or 24).

If the requesting host does not know its own IP address, it may leave the source field zero; the reply should then be broadcast. Since there is only one possible address format for a network, there is no need to match requests with replies. However, this approach should be avoided if at all possible, since it increases the superfluous broadcast load on the network.

Type A1 may be received from a gateway or a host.

Type A2 may be received from a gateway, or a host acting in lieu of a gateway.

II. Examples

For these examples, we assume that the requesting host has address 36.40.0.123, that there is a gateway at 36.40.0.62, and that on network 36.0.0.0, an 8-bit wide subnet field is in use.

First, suppose that broadcasting is allowed, and that 36.40.0.123 knows its own address. It sends the following datagram:

Source address:	36.40.0.123
Destination address:	36.255.255.255
Protocol:	ICMP = 1
Type:	Address Format Request = A1
Code:	0

36.40.0.62 will hear the datagram, and should respond with this datagram:

Source address:	36.40.0.62
Destination address:	36.40.0.123
Protocol:	ICMP = 1
Type:	Address Format Reply = A2
Code:	8

For the following examples, assume that address 255.255.255.255 denotes "broadcast to this physical network", as described in [6].

The previous example is inefficient, because it potentially broadcasts the request on many subnets. The most efficient method, and the one we recommend, is for a host to first discover its own address (perhaps using the "Reverse ARP" protocol described in [4]), and then to send the ICMP request to 255.255.255.255:

Source address:	36.40.0.123
Destination address:	255.255.255.255
Protocol:	ICMP = 1
Type:	Address Format Request = A1
Code:	0

The gateway can then respond directly to the requesting host.

Suppose that 36.40.0.123 is a diskless workstation, and does not know even its own host number. It could send the following datagram:

Source address:	0.0.0.0
Destination address:	255.255.255.255
Protocol:	ICMP = 1
Type:	Address Format Request = A1
Code:	0

36.40.0.62 will hear the datagram, and should respond with this datagram:

Source address:	36.40.0.62
Destination address:	36.40.255.255
Protocol:	ICMP = 1
Type:	Address Format Reply = A2
Code:	8

Note that the gateway uses the narrowest possible broadcast to reply (i.e., sending the reply to 36.255.255.255 would mean that it is transmitted on many subnets, not just the one on which it is needed.) Even so, the overuse of broadcasts presents an unnecessary load to all hosts on the subnet, and so we recommend that use of the "anonymous" (0.0.0.0) source address be kept to a minimum.

If broadcasting is not allowed, we assume that hosts have wired-in information about neighbor gateways; thus, 36.40.0.123 might send this datagram:

Source address:	36.40.0.123
Destination address:	36.40.0.62
Protocol:	ICMP = 1
Type:	Address Format Request = A1
Code:	0

36.40.0.62 should respond exactly as in the previous case.

Notes

<1> For example, some host have addresses assigned by concatenating their Class A network number with the low-order 24 bits of a 48-bit Ethernet hardware address.

<2> Our discussion of Internet broadcasting is based on [6].

<3> If broadcasting is not supported, them presumably a host "knows" the address of a neighbor gateway, and should send the ICMP to that gateway.

<4> This is what was referred to earlier as the coexistence of transparent and explicit subnets on a single network.

References

1. D.R. Boggs, J.F. Shoch, E.A. Taft, and R.M. Metcalfe. "Pup: An Internetwork Architecture." IEEE Transactions on Communications COM-28, 4, pp612-624, April 1980.

2. David D. Clark. Names, Addresses, Ports, and Routes. RFC-814, MIT-LCS, July 1982.

3. Yogan K. Dalal and Robert M. Metcalfe. "Reverse Path Forwarding of Broadcast Packets." Comm. ACM 21, 12, pp1040-1048, December 1978.

4. Ross Finlayson, Timothy Mann, Jeffrey Mogul, Marvin Theimer. A Reverse Address Resolution Protocol. RFC-903, Stanford University, June 1984.

5. R.M. Metcalfe and D.R. Boggs. "Ethernet: Distributed Packet Switching for Local Computer Networks." Comm. ACM 19, 7, pp395-404, July 1976. Also CSL-75-7, Xerox Palo Alto Research Center, reprinted in CSL-80-2.

6. Jeffrey Mogul. Broadcasting Internet Datagrams. RFC-919, Stanford University, October 1984.

7. David Plummer. An Ethernet Address Resolution Protocol. RFC-826, Symbolics, September 1982.

8. Jon Postel. Internet Protocol. RFC-791, USC-ISI, September 1981.

9. Jon Postel. Internet Control Message Protocol. RFC-792, USC-ISI, September 1981.